FAQ for the Microsoft Shared Source CLI Implementation

21 June 2001

What is the Microsoft Shared Source CLI Implementation?

In October of 2000, Microsoft and co-sponsors Hewlett Packard and Intel submitted specifications for the core of the .NET Framework and the C# programming language to ECMA, a well-known international standards association. On June 27, 2001, Microsoft announced that it is building a freely available, shared source implementation of the working drafts for these standards. This implementation is intended to meet the needs of academics, researchers, curious tinkerers, and those who wish to build independent versions of the proposed ECMA standards. We intend for it to be a fully functional, illustrative, implementation that can be used as a basis for non-commercial experimentation, as courseware, or as a guide for the use of parties working on their own commercial ECMA implementations.

The Microsoft Shared Source CLI Implementation will contain the source code to a complete implementation of the "common language infrastructure" (often referred to as the "CLI"), as well as two compilers, one for the C# language, and the other for the ECMAScript language. This CLI project shares much of its implementation with Microsoft's commercial CLI implementation, called the Microsoft .NET Framework, which is currently in beta.

For details on C#, ECMAScript, and the CLI, see http://msdn.microsoft.com/net/ecma/ or one of the many excellent books on these subjects.

Is this "Rotor?"

The internal codename for the project building the Microsoft Shared Source CLI implementation is "Rotor."

When will this CLI implementation be available?

We expect to have a beta-quality source code release will be available by the end of 2001 for both the Windows platform and for FreeBSD. The implementation is designed for use on PC class hardware.

Under what licensing terms will this CLI implementation be available?

The license for this source code will be available at the time of first public release. Microsoft intends to provide very liberal non-commercial licensing terms, and is interested in gathering community input on the design of the license. In particular, we intend to make it very easy for people to create non-commercial derivatives for exploration and experimentation, and for teaching purposes. We also intend to permit commercial use of this implementation as a guide for people building their own CLI implementations, for personal use, and for debugging purposes. Anyone expecting to use this implementation as the basis for distributing a commercial product would need to negotiate a license for this purpose with Microsoft.

Why FreeBSD?

One goal of creating a shared source implementation of the ECMA CLI is to prove that the technical choices being made by the ECMA technical group can be implemented on multiple operating system platforms. FreeBSD seemed like a good choice, since it is both a representative UNIX implementation and a platform that has historically encouraged unencumbered experimentation. Microsoft has no plans for supporting other platforms or chip architectures in this implementation at this time.

Who is working on this project? Can I participate?

Microsoft hopes to help foster many communities around the Shared Source implementation of the CLI. We are working on the details of this, and would appreciate input on possible approaches to community building and maintenance.

What is the relationship between this project and the Microsoft .NET Framework?

The Microsoft .NET Framework and its accompanying C# compiler are a commercial product, and have features not found in the ECMA working drafts. The center of gravity for this project is the working drafts themselves, and as such, is both smaller in footprint and easier to approach than the commercial product. The source code to the .NET Framework will be available under Microsoft's Shared Source Licensing Framework – see http://www.microsoft.com/sharedsource for more details.

What is the relationship between this project and the Microsoft .NET Compact Framework?

The Microsoft .NET Compact Framework is another implementation of the ECMA CLI, one that is specialized for small devices. The Microsoft .NET Compact Framework will have a subset of the overall .NET class libraries (in order to keep the size down) and it will also include device specific features like wireless APIs. This framework is being carefully designed to work well under the demanding constraints found on portable devices.